UNIVERSITY OF BIRMINGHAM

Information Extraction from Seminar Emails -

T.A. Goodman - **1526322**

A brief discussion on the task of Information Extraction from Seminar Emails, and analysis of the resulting solution to the problem, created using Python 3 and the nltk toolkit.

1 Outlining the Task

The given task was to use 300 tagged seminar articles to train a tagger that would then be able to tag 300 untagged seminar articles accurately in the same way that the tagged journals were tagged. The corpus was tagged specifically with paragraph, sentence, etime, stime, speaker and location tags, and similarly, tags in this form would have to be added to the untagged data

2 Preprocessing

Immediately after loading each individual file, it was important to split the file into three sections: the header, the 'Abstract:' line and the rest of the abstract itself, as each section needed to be treated differently. This was done by splitting on the regex "[Aa]bstract $\backslash s*$".

3 Isolating Paragraphs and Sentences

In the given corpus, paragraphs could be identified as being separated by two newline characters. Hence, obtaining a list of paragraphs was a simple as

$$data.split("\backslash n\backslash n")$$

The punkt tokenizer[1] provided by the nltk toolkit was instrumental in taking these paragraphs and splitting them into sentences. First, a tokenizer was initialised and then each paragraph was tagged by mapping the tokenize() function over the paragraphs with

$$list(map(sent\_detector.tokenize, paragraphs))$$

4 Information Extraction from the Header

In order to gain more information about the speaker, as well as the start and end times, the header was a valuable source of information. It was possible to extract such information with regex in most cases. Firstly the header was split in to lines (by splitting on \n) in order to circumvent some difficulties with multiline regex in Python.

4.1 Extracting the Start and End Times

There are many different formats that a time can appear in. Firstly, it could potentially contain either a start time, or both a start and an end time - i.e 10 $\vee$ 10-11. Moreover, there are a few other cases to take into account: Cases with AM/PM (10 10pm $\vee$ 10am), Minutes or no Minutes (10 $\vee$ 10:00), and combinations of these. In order to match all permutations of times, two regex expressions are required. One for the start time case,

$$([0-9]\{1,2\}\ ?([0-9]\{2\})?(([Aa]|[Pp])[Mm])?)$$

and one for the case of a start time followed by a hyphen, followed by an end time,

$$([0-9]\{1,2\}\ ?([0-9]\{2\})?\backslash s*(([Aa]|[Pp])[Mm])?)\backslash W*-\backslash W*([0-9]\{1,2\}\ ?([0-9]\{2\})?\backslash s*(([Aa]|[Pp])[Mm])?)$$

In order to ensure that the times obtained were definitely the start and end times (and not, for example, the sending time of the email), the times were only extracted from lines in the header that matched against

$$"[TtWw][IiHh][MmEe][EeNn]\backslash :([\backslash W\backslash w]*)"$$

5 Extracting the Speaker

Extraction of the speaker was very similar to the extraction of the start and end times for the most part. Firstly, a line was found that matched the regex

$$[Ww][Hh][Oo]\backslash :([\backslash W\backslash w]*)|[Ss][Pp][Ee][Aa][Kk][Ee][Rr]\backslash :([\backslash W\backslash w]*)$$

The captured group from this expression contains the speaker, but is not necessarily just the speaker, meaning that some further refinement has to be done - The first step of which is to remove leading spaces. The speaker could now feasibly be in the form "Dr. Mark Lee, Deputy Head of School, Head of Student Development Support". This is clearly not just the speaker, but also titles et al.

First, the phrase is tagged with the Named Entity Recognition (NER) tagger trained on the Wall Street Journal (WSJ) et al., and if an

$$<ENAMEX\ TYPE="PERSON"><\!/ENAMEX>$$

tag is present, the tagged phrase is used as the speaker. If this doesn't work, the following method is applied.

In order to remove the trailing titles and positions, a takeWhile function was defined, and $takewhile(phrase, punctuation)$ was performed, leaving the phrase as "Dr. Mark Lee". Moreover, the leading title still needs to be removed - in order to do this, the name was first converted to a list, i.e. ["$Dr.$","$Mark$","Lee"], and then all known leading titles (Eg. Dr, Prof, Mr, Mrs...) were removed from the list, leaving ["$Mark$","Lee"], which can then be interspersed with spaces (["$Mark$","","Lee"]) and joined on to an empty string, giving "Mark Lee".

5.1 Extracting the Location

Due to time constraints, extraction of location wasn't implemented, but would be extremely similar to the previous two methods of extraction.

6 Evaluation of the Tagger

In order to evaluate the tagger, it was measured on it's ability to accurately tag speaker, stime and etime, as well as the overall accuracy. Because of oversights in the evaluation code and time constraints, it does not accurately calculate the accuracy of paragraph and speaker tagging. Moreover, the actual results are likely to be much higher than the calculated results. Through manual observation, the tagger is consistently accurate across the untagged data.

Tag	**Accuracy**
speaker	28.09%
stime	15.24%
etime	49.44%

giving an overall accuracy of 26.99%.

7 Future Considerations & Conclusion

Given more time, the first priority would be to implement the location tagging of the data. Following that, using chunking on a wider scale to identify Named Entities (or using the NER tagger on the whole data set) would likely give rise to more accurate tagging overall.

It would also be imperative to vastly improve the evaluation code in order to make sure that it gave accurate and representative results based on the system.

All in all, there is a lot that could be improved with the current system, but it is definitely a strong basis for a better tagger, as is evidenced by it's ability to tag the seminar information relatively consistently across the whole dataset.

References

[1] http://www.nltk.org/api/nltk.tokenize.html